

# NAG Library Function Document

## nag\_zload (f16hbc)

### 1 Purpose

nag\_zload (f16hbc) broadcasts a scalar into a complex vector.

### 2 Specification

```
#include <nag.h>
#include <nagf16.h>

void nag_zload (Integer n, Complex alpha, Complex x[], Integer incx,
               NagError *fail)
```

### 3 Description

nag\_zload (f16hbc) performs the operation

$$x \leftarrow (\alpha, \alpha, \dots, \alpha)^T,$$

where  $x$  is an  $n$  element complex vector and  $\alpha$  is a complex scalar.

### 4 References

The BLAS Technical Forum Standard (2001) <http://www.netlib.org/blas/blast-forum>

### 5 Arguments

- |    |   |                     |
|----|---|---------------------|
| 1: | <b>n</b> – Integer  | <i>Input</i>        |
|    | <i>On entry:</i> $n$ , the number of elements in $x$ .  |                     |
|    | <i>Constraint:</i> $n \geq 0$ .   |                     |
| 2: | <b>alpha</b> – Complex  | <i>Input</i>        |
|    | <i>On entry:</i> the scalar $\alpha$ .  |                     |
| 3: | <b>x</b> [ <i>dim</i> ] – Complex   | <i>Output</i>       |
|    | <b>Note:</b> the dimension, <i>dim</i> , of the array <b>x</b> must be at least $\max(1, 1 + (n - 1) incx )$ .                |                     |
|    | <i>On exit:</i> the scalar $\alpha$ scattered with a stride of <b>incx</b> . Intermediate elements of <b>x</b> are unchanged. |                     |
| 4: | <b>incx</b> – Integer   | <i>Input</i>        |
|    | <i>On entry:</i> the increment in the subscripts of <b>x</b> between successive elements of $x$ .                             |                     |
|    | <i>Constraint:</i> <b>incx</b> $\neq 0$ .   |                     |
| 5: | <b>fail</b> – NagError *  | <i>Input/Output</i> |
|    | The NAG error argument (see Section 3.6 in the Essential Introduction).   |                     |

### 6 Error Indicators and Warnings

#### NE\_BAD\_PARAM

On entry, argument  $\langle value \rangle$  had an illegal value.

**NE\_INT**

On entry, **incx** =  $\langle value \rangle$ .  
 Constraint: **incx**  $\neq 0$ .

On entry, **n** =  $\langle value \rangle$ .  
 Constraint: **n**  $\geq 0$ .

**7 Accuracy**

Not applicable.

**8 Further Comments**

None.

**9 Example**

The scalar  $0.5 - 0.3i$  is loaded into a vector of length 4, stored in **x** with increment 2 (**incx** = 2).

**9.1 Program Text**

```

/* nag_zload (f16hbc) Example Program.
 *
 * Copyright 2005 Numerical Algorithms Group.
 *
 * Mark 8, 2005.
 */

#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagf16.h>
#include <nagx04.h>

int main(int argc, char *argv[])
{
    FILE      *fpin, *fpout;

    /* Scalars */
    Complex  alpha;
    Integer  exit_status, i, incx, n, xlen;

    /* Arrays */
    Complex  *x = 0;

    /* Nag Types */
    NagError fail;

    exit_status = 0;
    INIT_FAIL(fail);

    /* Check for command-line IO options */
    fpin = nag_example_file_io(argc, argv, "-data", NULL);
    fpout = nag_example_file_io(argc, argv, "-results", NULL);

    fprintf(fpout, "nag_zload (f16hbc) Example Program Results\n\n");

    /* Skip heading in data file */
    fscanf(fpin, "%*[\n] ");

    /* Read length of vector and increment. */
    fscanf(fpin, "%ld%ld%*[\n] ", &n, &incx);

    /* Read scalar parameter */
    fscanf(fpin, " ( %lf , %lf ) %*[\n] ", &alpha.re, &alpha.im);

```

```

xlen = MAX(1, 1 + (n - 1)*ABS(incx));
if (n > 0)
{
    /* Allocate memory */
    if (!(x = NAG_ALLOC(xlen, Complex)))
    {
        fprintf(fpout, "Allocation failure\n");
        exit_status = -1;
        goto END;
    }
}
else
{
    fprintf(fpout, "Invalid n\n");
    exit_status = 1;
    return exit_status;
}

/* nag_zload(f16hbc).
 * Broadcast a complex scalar to a complex vector.
 *
 */
nag_zload(n, alpha, x, incx, &fail);
if (fail.code != NE_NOERROR)
{
    fprintf(fpout, "Error from nag_zload.\n%s\n", fail.message);
    exit_status = 1;
    goto END;
}

/* Print x. */
fprintf(fpout, "Loaded vector x:\n\n");
for (i = 0; i < xlen; i = i + incx)
    fprintf(fpout, " x[%11d] = (%5.2f, %5.2f)\n", i, x[i].re, x[i].im);
END:
if (fpin != stdin) fclose(fpin);
if (fpout != stdout) fclose(fpout);
if (x) NAG_FREE(x);

return exit_status;
}

```

## 9.2 Program Data

nag\_zload (f16hbc) Example Program Data

4	2	:	n, incx the length and increment of x
( 0.5,-0.3)		:	alpha

## 9.3 Program Results

nag\_zload (f16hbc) Example Program Results

Loaded vector x:

```

x[0] = ( 0.50, -0.30)
x[2] = ( 0.50, -0.30)
x[4] = ( 0.50, -0.30)
x[6] = ( 0.50, -0.30)

```

---